April 1993 UILU-ENG-93-2210

NEW SCHEMES FOR PREDICTIVE CONGESTION CONTROL

Bevan Narayan Das

Coordinated Science Laboratory
College of Engineering
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

NEW SCHEMES FOR PREDICTIVE CONGESTION CONTROL

BY

BEVAN NARAYAN DAS

B.S., Cornell University, 1990
A.B., Cornell University, 1990

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1993

Urbana, Illinois

REPORT DOCUMENTATION PAGE

Form Approved

AFRL-SR-BL-TR-98-

0443

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYP
	April 1993	Final

4. TITLE AND SUBTITLE
New Schemes for Predictive Congestion Control

5. FUNDING NUMBERS

6. AUTHORS
Bevan Narayan Das

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Illinois at Urbana-Champaign

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NI
110 Duncan Avenue, Room B-115
Bolling Air Force Base, DC 20332-8080

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*
See attached.

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

14. SUBJECT TERMS	15. NUMBER OF PAGES
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

ABSTRACT

The Predictive Congestion Control framework, as proposed by Ko, Mishra, and Tripathi [1], applies to high-speed, wide-area communication networks. The central assumption of the framework is that the link propagation delay, a deterministic quantity depending on fiber length, dominates all other types of delay in the networks. Within the framework, the authors developed four schemes, the static, optimistic, pessimistic, and heuristic, which are summarized here. This thesis presents two new schemes and compares their performances to those of the original four schemes. One new scheme, the square-root queue scheme, adapted from a dynamic window scheme of Mitra and Seery [3], attempts to equate the square of the average buffer occupancy and the product of the output rate and the link propagation delay. The second new scheme, the equal-risk principle scheme, motivated by keeping the probability of packet loss small, compares available space to the requested output rate. The schemes were tested on two cases in simulations: one virtual circuit with cross-traffic sharing one link, and two virtual circuits having one link in common. The measures used to judge performance are packet-loss ratios for the congested link, for the internal network, and for the end-to-end virtual circuit. Based on these measures, the square-root queue scheme and the equal-risk scheme perform as well as but not significantly better than the static scheme.

ACKNOWLEDGMENTS

I thank my advisor Bruce Hajek for his several helpful ideas and discussions during the course of this work. I also want to recognize the financial support from the Department of Defense Science and Engineering Graduate Fellowship.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In this thesis, the Predictive Congestion Control framework described by Ko, Mishra, and Tripathi for high-speed, wide-area networks is explored. In [1], they assume that with fiber optic links and with advances in switching technology, transmission errors–due to noisy carriers–and switching delays will be greatly reduced. Furthermore, as integrated circuit speeds increase, the corresponding processing delay for packets will decrease. Thus, in the Predictive Congestion Control framework, instead of four causes of delay–processing, switching, propagation, and queueing–there are only two–propagation and queueing. Since the speed of light is a limit, and since distances between communicating nodes are known, the propagation delay is assumed to be deterministic. The other source of delay, queueing, is determined by the specific control scheme used in the network.

In [1, 2], Ko et al. examine four schemes–static, optimistic, pessimistic, and heuristic–in a variety of network configurations and conditions. In this thesis, two more schemes are devised and examined–the square-root queue and the equal-risk principle schemes–and their performances are compared to those of the original schemes.

The main motivation for predictive congestion control[1] is to use the fact that (1) propagation and processing delays in high-speed networks are nearly deterministic, and (2) accurate knowledge of propagation delays enables nodes to accurately estimate each other's states. By assuming that the delay is dominated by the propagation delay, the time "distance" between nodes can be assumed to be known. Node i, upon receiving information from a neighbor, e.g., node j, can mark that information with a specific age,

[1]When referring to the specific framework of Ko, Mishra, and Tripathi, "Predictive Congestion Control" is written with capitalized letters; while referring to the general concept of congestion control that estimates future behavior of the network, the term "predictive congestion control" is not capitalized.

as in "This information is D_{ij} time units old," where D_{ij} is the known delay between nodes i and j.

Furthermore, the control scheme at node i can estimate the effect of a control decision at i upon a future state of the neighbor j. The state of a node could be the buffer occupancy at the node and the input and output rates of virtual circuits passing through the node.[2] With the link delays known and deterministic, Ko, Mishra, and Tripathi argue that the estimates of the neighboring states based on the deterministic propagation delays are better than estimates based on random link delays.

To take advantage of the known link delays, Ko et al. proposed the Predictive Congestion Control framework. In this framework, node i, at various control decision times, adjusts the output rates of virtual circuits passing through node i. The decisions depend on the current state of node i and the estimated future states of node i's neighbors. Node i can use estimates of other nodes to influence its decisions, but utilizes only information passed to it from its neighbors, so that it knows the age of the information. For example, at time t, node k sends information to its neighbor node j; node j receives the information at $t + D_{jk}$. Node j can then send node k's information to node i; node i receives the information at $t + D_{ij} + D_{jk}$ (or at $t + D_{ij} + \Delta_j + D_{jk}$, where Δ_j is the length of time that node j holds on to node k's information before sending on to node i). The quantity $D_{ij} + D_{jk}$ (or $D_{ij} + \Delta_j + D_{jk}$) is the age of the information.

This framework requires the use of decentralized control. Output rates are adjusted hop-by-hop. Information at each node concerns only that node and its neighbors, or it is global information filtered by the neighbors. For example, if $i \rightarrow j \rightarrow k$ is the path from node i to node k, for example, then node i can eventually know D_{ij} and D_{jk} , but the delay from node i to node k is not merely the sum of the two. The queueing delay at node j, a time-varying quantity, has to be added in. One advantage of this method is that changes in the network structure are localized; information must only be updated near the site of the structure change. However, to maintain this flexibility, the packets

[2]The Predictive Congestion Control framework assumes a virtual circuit-switched network.

(maybe only state information packets) require an "age" field, which would accumulate the various D_{ij}'s and Δ_j's as the packets travel through the network.

Ko, Mishra, and Tripathi leave the field open for specific control schemes to be used in the Predictive Congestion Control framework. They discuss four schemes in [1, 2]: static, pessimistic, optimistic, and heuristic.

CHAPTER 2

KO, MISHRA, AND TRIPATHI SCHEMES

For the schemes discussed below, Ko et al. assume that the links have uniform delays, that is, D_{ij} is identically equal to D, for all pairs of neighboring nodes i, j in the network.

In addition, Ko et al. assume that the output rate of a particular virtual circuit at a given node is constant between decision times. For this to happen, the system has to have "announced arrivals." For example, node i receives an announcement at decision time t that it will receive x packets during $[t, t + D)$ on virtual circuit V, where $t + D$ is the next decision time. Node i can use the announced value x in determining its own output rates over the same interval.

Ko et al. use two performance measures to judge the performance of each control scheme: average throughput and average delay (including only successfully transmitted packets). Though these two measures concern end-to-end performance, Ko et al. argue that in a hop-by-hop context, matching the input rates to the output rates at each node along the virtual circuits guarantees good performance as judged by the end-to-end throughput and delay.

2.1 Static Scheme

In this scheme, the output rates are selected at the start of operations and then held constant. If necessary, that is, if a virtual circuit is rerouted or a new virtual circuit is added, then the output rates may be adjusted.

To match rates, the output rate of each node for a specific virtual circuit is set to the mean rate of that virtual circuit's source. For example, in Figure 2.1, source S_i has mean μ_i. Hence, the static scheme assigns the following output rates $R_{node,output,VC\#}$:

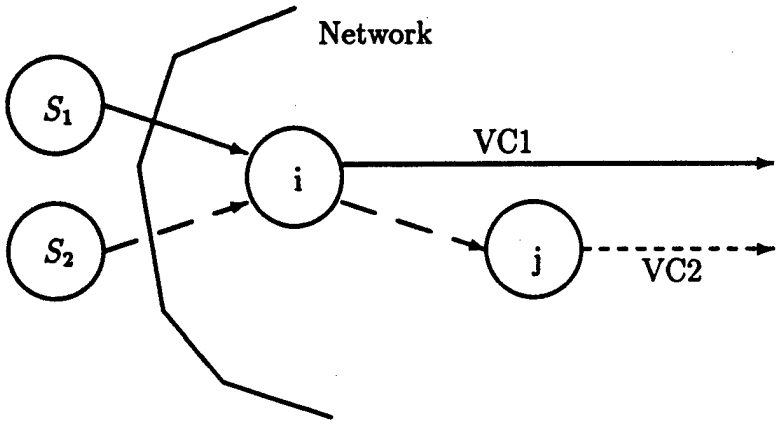

Figure 2.1 Network model with two virtual circuits

node i: $R_{i,out,1} = \mu_1$ $R_{i,out,2} = \mu_2$ total$R_{i,out} = \mu_1 + \mu_2$

node$j$: $R_{j,out,2} = \mu_2$ total$R_{j,out} = \mu_2$

Note that the static scheme is not specifically a predictive congestion control scheme. It is included to provide a benchmark for comparison. If dynamic control schemes (such as those below) cannot perform better than the static scheme (with fixed parameters) over a broad range of traffic patterns, then there is no need to bother with estimating the future states of the neighboring nodes.

2.2 Optimistic Scheme

In this scheme, the nodes in the network assume that the downstream nodes can handle any traffic passed down to them. Roughly speaking, each node matches input and output rates by explicitly setting its output rate for a given virtual circuit equal to the input rate for that same circuit. Specifically, the output rate for a given node and virtual circuit at a given decision time is set to the flushing rate, which is the sum of the input rate plus any additional rate needed to empty the buffer by the next decision time. But, if the initial buffer is empty, the buffer never fills up; hence, the flushing rate identically equals the input rate. There is a difference between the flushing rate and the input rate only if there is a failure somewhere. In that case, some node has a zero output rate, causing its buffers to fill up.

2.3 Pessimistic Scheme

2.3.1 Description of the scheme

The main goal of the pessimistic scheme is to never allow the downstream neighbor's buffers to overflow. That is, the node does not send a packet unless it knows that there will be space for the packet in the next node's buffers.

To describe this scheme in detail, the following notation is introduced:

$B_i(t)$ = buffer occupancy of node i at time t

$R_{i,in}(t)$ = input rate of node i at time t

$R_{i,out}(t)$ = output rate of node i at time t

This notation deals with only one virtual circuit at a time; when there is more than one virtual circuit, the subscript includes an additional V term, where V is the identifier of the specific virtual circuit. For example, $B_{1,2}(t)$ is the buffer occupancy of virtual circuit 2's packets at node 1 at time t.

A difference between this notation and the notation of Ko, Mishra, and Tripathi is the use of $B_i(t)$. Ko et al. use $B_i(t)$ to denote the amount of free buffer space at node i at time t.[1] For this analysis of their schemes, $B_i(t)$ denotes the amount of buffer space occupied at node i at time t, because it is easier to think in terms of how many packets are in the buffer, not in terms of how many packets are not there. To convert from one to the other, $B_i(t)_{Ko} = B_{\max} - B_i(t)_{Das}$. One advantage of the approach in this thesis is in the case of different $B_{\max}$'s.

If decision times are every D time units, then node i attempts to set $R_{i,out}(t)$ so that

$$B_j(s) \leq B_{\max}, t + D \leq s \leq t + 2D$$

[1]See, for example, p. 12 of [1].

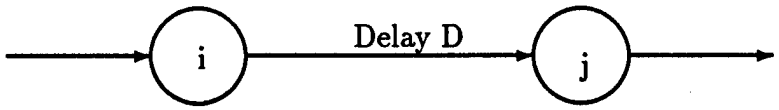

Figure 2.2 Nodes i and j with corresponding delay

where j is i's downstream neighbor[2] and $B_{\max}$= maximum buffer occupancy per VC per node (same at all nodes) (see Figure 2.2).

Since $R_{i,out}(\cdot)$ and $R_{j,out}(\cdot)$ are constant between decision times,

$$B_j(t+D) = B_j(t) + [R_{j,in}(t) - R_{j,out}(t)] * D.$$

By rescaling time, it can be assumed that $D = 1$; thus,

$$B_j(t+1) = B_j(t) + [R_{j,in}(t) - R_{j,out}(t)].$$

Next, from the relationship $R_{j,in}(t) = R_{i,out}(t-1)$, the equation becomes

$$B_j(t+1) = B_j(t) + [R_{i,out}(t-1) - R_{j,out}(t)]$$

and similarly for the $B_j(t+2)$

$$B_j(t+2) = B_j(t+1) + [R_{i,out}(t) - R_{j,out}(t+1)].$$

Substituting in for $B_j(t+1)$ yields

$$B_j(t+2) = B_j(t) + R_{i,out}(t-1) - R_{j,out}(t) + [R_{i,out}(t) - R_{j,out}(t+1)].$$

The goal is to have $B_j(t+2) \leq B_{\max}$, which requires

$$R_{i,out}(t) \leq B_{\max} - B_j(t) - R_{i,out}(t-1) + R_{j,out}(t) + R_{j,out}(t+1).$$

At time t, node i knows $R_{i,out}(t-1)$ and can calculate $B_j(t)$ from the known quantities $B_j(t-1)$, $R_{j,in}(t-1)$, and $R_{j,out}(t-1)$. The node does not know $R_{j,out}(t)$ nor

[2]The notation in this thesis for node i's downstream neighbor depends on the context. When the nodes are numbered sequentially, the downstream node is marked $i+1$, while for general networks, the downstream neighbor has the generic label j.

$R_{j,out}(t+1)$; hence, the pessimistic scheme makes a worst-case assumption that $R_{j,out}(t) = R_{j,out}(t+1) = 0$. Hence the constraint takes the form

$$R_{i,out}(t) \leq B_{\max} - B_j(t) - R_{i,out}(t-1).$$

Finally, taking into account the input rate, the control law is

$$R_{i,out}(t) = \min(\text{opt}, \text{pess}) \tag{2.1}$$

where opt $= R_{i,in}(t) + B_i(t)$ and pess $= B_{\max} - B_j(t) - R_{i,out}(t-1)$.

(With the same notation, the optimistic scheme can be represented as $R_{i,out}(t) =$ opt .)

The memory at each node has separate buffers for each virtual circuit passing through it. The maximum buffer space $B_{\max}$ is per virtual circuit. For example, if three virtual circuits pass through node i, then node i will have $3B_{\max}$ buffer spaces reserved.

2.3.2 Oscillation in pessimistic scheme

Ko, Mishra, and Tripathi remark in [1] that the size of $B_{\max}$ affects the performance of the pessimistic scheme. Setting the maximum buffer occupancy too small may lead to excessive oscillations. They also note in [1] that, under the pessimistic scheme, the only packets lost are dropped at the first node; no packets are dropped after the first node. To illustrate these points more clearly, the discussion below is included.

In Figure 2.3, there is a virtual circuit V passing through nodes 1,2,3,... successively. (There may be more nodes in V, but only nodes 1, 2, and 3 are of concern here.) Each link has the capacity of 1 gigabit per second (Gb/s), and each node has up to $B_{\max}$ buffer slots available for V's packets. Each packet is 4 K, or 32768 bits, long, and the one-way propagation delay between each neighboring pair of nodes is 10 ms. Hence, each link can pass up to 304 packets per delay time period. The source S offers Poisson traffic with mean 500 megabits per second (Mb/s), or, equivalently, 152 packets per delay time period.

Tables 2.1 and 2.2 present the numbers of packets in each buffer or each link at the decision times $t = 0, 1, 2, \ldots$, (where the time unit is one delay period $D = 10$ ms) for the

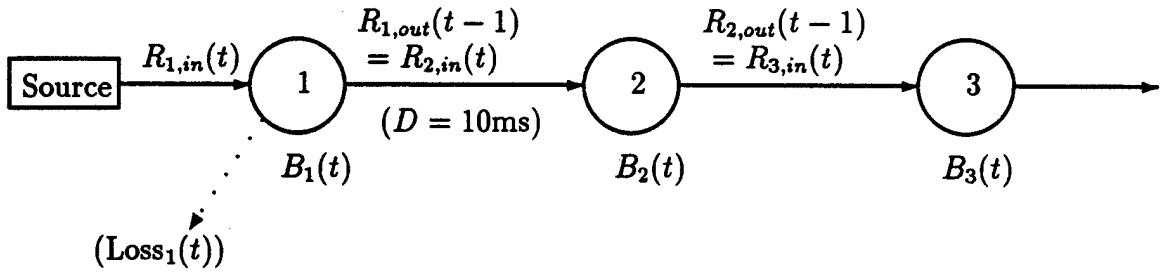

Figure 2.3 Virtual circuit for illustration of oscillation with pessimistic scheme

Table 2.1 Pessimistic scheme with $B_{max} = 100 = \frac{1}{3}$ capacity, mean $= 152 = \frac{1}{2}$ capacity

t	$R_{1,in}(t)$	$(\text{Loss}_1(t))$	$B_1(t)$	$R_{2,in}(t)$	$B_2(t)$	$R_{3,in}(t)$	$B_3(t)$
0	148						
1	153	(0)	48	100	0		
2	156	(101)	100	0	0	100	0
3	151	(56)	100	100	0	0	0
4	149	(151)	100	0	0	100	0
5	157	(49)	100	100	0	0	0
6	152	(157)	100	0	0	100	0
7	147	(52)	100	100	0	0	0

pessimistic control scheme in two cases. The first case has $B_{\max}$= 100 packet slots, the second case has $B_{\max} = 2\lambda D$, where λ is the mean rate for the VC source. The formulas for deriving the numbers in the table are summarized below. The following equations hold for all $t = 1, 2, 3, \ldots, i = 1, 2, 3, \ldots$:

$$B_i(t) = \min(B_i(t-1) + R_{i,in}(t-1) - R_{i,out}(t-1), B_{\max}) \tag{2.2}$$

$$R_{i,out}(t) = \min(\text{opt}, \text{pess}) \tag{2.3}$$

$$\text{Loss}_1(t) = [B_i(t-1) + R_{1,in}(t-1) - R_{1,out}(t-1) - B_{\max}]^+ \tag{2.4}$$

$$R_{i,out}(t) = R_{i+1,in}(t+1) \tag{2.5}$$

where opt $= R_{i,in}(t) + B_i(t)$ and pess $= B_{\max} - B_{i+1}(t) - R_{i,out}(t-1)$.

Note that (2.5) implies that $R_{i,out}(t)$ is actually on the row for time $t+1$ in the tables (under the heading $R_{i+1,in}(t+1)$). Since $R_{i,out}(t)$ is the output rate for $[t, t+1)$, the packets sent out from node i at this rate are spread evenly along link $(i, i+1)$ at time $t+1$. (Because a time unit equal to one propagation delay D is used, the number of

Table 2.2 Pessimistic scheme with $B_{max} = 304$ = capacity, mean = 152 = $\frac{1}{2}$ capacity

t	$R_{1,in}(t)$	(Loss$_1(t)$)	$B_1(t)$	$R_{2,in}(t)$	$B_2(t)$	$R_{3,in}(t)$	$B_3(t)$
0	148						
1	153	(0)	0	148	0		
2	156	(0)	0	153	0	148	0
3	151	(0)	5	151	0	153	0
4	149	(0)	3	153	0	151	0
5	157	(0)	1	151	0	153	0
6	152	(0)	5	153	0	151	0
7	147	(0)	6	151	0	153	0

packets transmitted by node i during $[t, t+1)$ is the output rate multiplied by 1; instead of writing $R_{i,out}(t) * 1$, $R_{i,out}(t)$ is used for both the output rate and the number of packets transmitted.)

For an example of the calculations, with $B_{\max} = 100$ in Table 2.1, consider the line for $t = 2$. The quantity $R_{1,in}(2)$ is the number of packets output from the source during the time period [2,3); this number is Poisson(152). The number 156 is only one of several possible values. Since $B_1(1) = 48$, $R_{1,in}(1) = 153$, and $R_{1,out}(1) = 0$, by (2.2) and (2.4)

$$B_1(2) = \min(48 + 153 - 0, 100) = 100$$

$$\text{Loss}_1(2) = [(48 + 153 - 0) - 100]^+ = 101$$

$$B_j(2) = 0, j = 2, 3, \ldots \tag{2.6}$$

$$\text{Loss}_j(2) = 0, j = 2, 3, \ldots,$$

where $B_j(2) = \text{Loss}_j(2) = 0$ because of the pessimistic scheme. Next, from (2.3) and (2.6), (for the next line $t = 3$)

$$R_{1,out}(2) = R_{2,in}(3) = \min(201, 100) = 100$$

$$R_{2,out}(2) = R_{3,in}(3) = \min(0, 0) = 0.$$

The tables illustrate how the output rates oscillate in relation to the size of $B_{\max}$. Furthermore, the lack of oscillations in Table 2.2 suggests that the threshold for cutting off oscillations may be $B_{\max}$= twice the mean rate of the source multiplied by the link propagation delay (equivalent to a "round-trip window size").

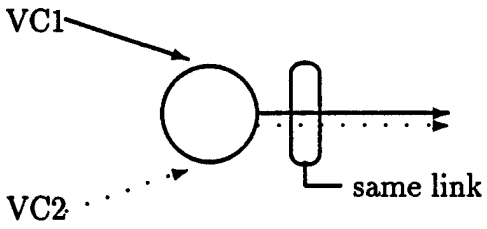

Figure 2.4 Two virtual circuits sharing a link

2.4 Heuristic Scheme

2.4.1 Motivation for heuristic scheme

Before discussing the fourth control scheme of Ko, Mishra, and Tripathi, the heuristic scheme, the effects of cross-traffic are considered. In the above three schemes, the cross-traffic does not affect the decision of the node concerning a specific virtual circuit. For example, consider Figure 2.4.

At node i, VC2 is cross-traffic for VC1 (and vice versa), but node i does not take VC2's traffic into account when deciding $R_{i,out,1}(t)$ if the static, optimistic, or pessimistic schemes are employed.

The effects of cross-traffic are felt only after the decision is made. For example, if the sought-after values of VC1's output rate and VC2's output rate at node i at time t add up to more than the capacity of the shared outgoing link, either one or both of the virtual circuits do not receive the full bandwidth that they request. If, for example, VC1 has priority over VC2, then VC1 most likely receives all of its requested output rate, and VC2 receives whatever capacity is left over. If, however, the two virtual circuits share the same priority, some other, fairer allocation is needed.

To differentiate between the rate requested and what is actually allocated, the notations "requested output rate" and "actual output rate" are used. The value $R_{i,out,VC}(t)$ determined by the static, optimistic, and pessimistic schemes should be viewed as the output rate that the virtual circuit requests to receive, not as the output rate that the virtual circuit actually receives. What portion of the link capacity it actually receives

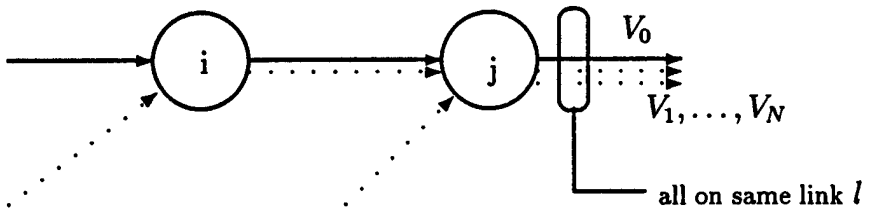

Figure 2.5 Virtual circuit example for heuristic scheme

depends on the total requested rate for that link and the allocation procedure used by the node.

2.4.2 Description of the heuristic scheme

The heuristic scheme explicitly takes into account the amount of cross-traffic when deciding the output rate at a given node for a given virtual circuit. The other three schemes (static, optimistic, and pessimistic) do not, as noted above. The heuristic scheme attempts to match the output rate of a given node to both the output rate of its upstream neighbor (at the previous decision time) and of its downstream neighbor (at the next decision time). If the (estimated) upstream and downstream rates do not match each other, an intermediate value is chosen as the requested output rate at the given node, and the choice takes into account cross-traffic rate values. In practice, the estimated downstream rate is first taken into account, then the upstream rate is taken into account.

To illustrate the detailed operation of the heuristic scheme, Figure 2.5 is presented.[3]

To decide $R_{i,out,V_0}(t)$, the heuristic scheme first estimates $\hat{R}_{j,out,V_0}(t+D)$ (Ko, Mishra, and Tripathi use a weighted moving-average estimator $\hat{R}(t) = \sum_{n=1}^{3} a_n R(t - nD), a_1 \geq a_2 \geq a_3 \geq 0, \sum a_n = 1$). The scheme then checks the utilization of the outgoing link (e.g., link l) from j that V_0 shares with $V_1, V_2, \ldots, V_N$. If this value, $\sum_{n=1}^{N} \hat{R}_{j,out,V_n}(t+D)$, is greater than some specified fraction of link l's capacity, then the heuristic scheme adds to $\hat{R}_{j,out,V_0}(t+D)$ a fair share of the remaining, unused capacity of link l. On the other hand, if the utilization of link l is below the specified fraction of l's capacity, then the

[3]The description does not include all of the heuristic scheme's equations in full detail. See [2] for the full description.

heuristic scheme adjusts $\hat{R}_{j,out,V_0}(t+D)$ depending on the fraction of buffer space at node j allotted to V_0 that is estimated to be occupied by V_0's packets at time $t+D$. If $\hat{B}_{j,V_0}(t+D)/B_{\max}$ is near 1, then $\hat{R}_{j,out,V_0}(t+D)$ is decreased; if the fraction is near 0, then a large portion of l's unused capacity is added to $\hat{R}_{j,out,V_0}(t+D)$; and if the fraction is neither near 0 nor 1, then a small portion of l's unused capacity is added to $\hat{R}_{j,out,V_0}(t+D)$. Finally, the adjusted value of $\hat{R}_{j,out,V_0}(t+D)$ is compared to the flushing rate (which depends on the upstream output rate). The actual output rate, $R_{i,out,V_0}(t)$, is the minimum of the two values. That is, similar to the pessimistic scheme, the heuristic control law has the form $R_{i,out,V_0}(t) = \min(\text{opt}, \hat{R}_{j,out,V_0}(t+D))$, where opt is the flushing rate, defined as above for the optimistic and pessimistic schemes.

The $\hat{}$ notation means that the marked quantity is an estimate, not an observed value. Note though that, in this case of node j being node i's downstream neighbor, $\hat{B}_j(t)$ is not written for the buffer occupancy of node j at time t (as estimated by node i), t a decision time. The calculated value of

$$B_j(t) = B_j(t-D) + [R_{j,in}(t-D) - R_{j,out}(t-D)] * D$$

follows completely from information known at node i at time t. Hence, while not a directly observed value, $B_j(t)$ is known to i at time t.

CHAPTER 3

NEW SCHEMES

3.1 Square-root Queue Scheme

In this section the first of two new Predictive Congestion Control schemes–the square-root queue scheme–is presented and described. First, a congestion control scheme described by Mitra and Seery is summarized and why this scheme appears suitable for the Predictive Congestion Control framework is pointed out. However, the framework on which Mitra and Seery design their scheme is different than the Predictive Congestion Control framework, and several adjustments have to be made before the scheme can be used in the Predictive Congestion Control framework. Out of these adjustments comes the development of a scheme similar to the original Mitra and Seery scheme; this scheme is the square-root queue scheme.

3.1.1 Motivation for a new scheme

In [1, 2], Ko, Mishra, and Tripathi argue that packet loss because of buffer overflow will be the main cause of unsuccessful transmissions (as opposed to transmission errors caused by a lossy transmission medium). Yet, the only scheme they propose which has no packet loss inside the network is the pessimistic scheme. The pessimistic scheme, however, requires excessively large buffers to avoid large oscillations in the network traffic. In the search for a control scheme that minimizes packet loss while needing minimal buffers, the dynamic window flow control proposed by Mitra and Seery in [3] appears to be a good candidate.

Mitra and Seery describe a scheme that works for high-speed, wide-area networks with propagation delays much larger than the other sources of delay. In their analysis

of such a network, they find an operating point for the control scheme that maximizes the power (= throughput/delay) of each virtual circuit. Since this operating point is in the region where the window size is approximately equal to the number of packets offered during one round-trip propagation delay, the operating point gives an optimal relationship between the window size and mean round-trip response time. The response time, which can be observed, can then be used to adjust the window size towards the optimal size. The window size-response time relationship is not linear, and it happens to offer an advantage similar to additive increase-multiplicative decrease window size control.

3.1.2 Summary of Mitra-Seery scheme

For a virtual circuit with M bottleneck nodes, Mitra and Seery present the following control scheme:[1] for packet $n, n = 1, 2, \ldots$, measure the round-trip response R_n, calculate the bias $B_n = (R_n - 1)\sqrt{K_n} - \sqrt{M}$, and adjust the window size $K_{n+1} = K_n - aB_n$. Mitra and Seery scale time so that one time unit equals the round-trip propagation delay, the quantity $R_n - 1$ denotes the (scaled) time that packet n spends waiting in queues between the source and the destination. Then, with R^* and K^* as optimal values for the response time and window size, respectively, Mitra and Seery note that $(R^* - 1)\sqrt{K^*} = b$. After both sides are multiplied by $\sqrt{K^*}$, this relationship can be interpreted as follows. The mean total number of packets waiting in queues at any given moment ($(R^* - 1)K^*$) is proportional to the square-root of the window size ($b\sqrt{K^*}$). Finally, the bias function follows from finding the constant of proportionality, $b = \sqrt{M}$.

Since the authors restrict the change in window size to the set $\{-1, 0, 1\}$, the process $(K_n)_{n \geq 0}$ represents a virtual window size; the actual window size $\tilde{K}_{n+1}$ changes according

[1]The scheme is only summarized here. To understand in more detail the choice of bias function and window control law, see [3].

to the rule

$$\tilde{K}_{n+1} = \begin{cases} \tilde{K}_n - 1 & \text{if } K_{n+1} \leq \tilde{K}_n - 1 \\ \tilde{K}_n & \tilde{K}_n - 1 \leq K_{n+1} \leq \tilde{K}_n + 1 \\ \tilde{K}_n + 1 & \tilde{K}_n + 1 < K_{n+1} \end{cases} .$$

This scheme has some advantages. For one, the virtual circuits that share links can have decoupled control. The queue lengths are only approximately the square-root of the number of packets offered in one round-trip propagation delay period, much smaller than the buffer sizes needed in the pessimistic scheme. And, most importantly, with high probability, there is no packet loss. Hence the scheme appears to be a good candidate for a Predictive Congestion Control scheme.

3.1.3 Adjustments necessary for Predictive Congestion Control framework

The differences between the framework that supports the Mitra and Seery scheme and the Predictive Congestion Control framework need to be resolved before the new scheme can be used as a Predictive Congestion Control scheme. On the one hand, the Mitra and Seery scheme is window-based, end-to-end control designed to handle data traffic. On the other hand, the Predictive Congestion Control framework supports rate-based, hop-by-hop control schemes for general traffic. In addition, the Mitra and Seery scheme uses acknowledgements in determining the mean round-trip response time, while acknowledgements do not exist in the Predictive Congestion Control framework.

Because of some assumptions by Ko, Mishra, and Tripathi about the four Predictive Congestion Control schemes that they develop, the differences mentioned above are not difficult to resolve. First, since the output rates are assumed to be piecewise constant, the rate-based control becomes equivalent to window-based control; the (constant) output rate multiplied by a length of time–decision interval or propagation delay–gives an equivalent window size. Second, to equate end-to-end control with hop-by-hop control, each link is viewed as a complete (virtual) circuit. That is, in Mitra and Seery's notation, M is set equal to one. Third, while developing their four specific schemes, Ko, Mishra, and Tripathi implicitly assume that the traffic is data-like. They model the sources as

Poisson distributed and they emphasize minimizing delay and maximizing throughput as performance criteria; these source models and performance criteria apply more to data traffic than do other types of traffic (voice and video, for example).

Finally, Mitra and Seery note that the design equation relating the response time to the window size is not the only basis for a control scheme; they state another design equation relating the average nodal queue size to the window size. While acknowledgements are not part of Predictive Congestion Control, the buffer occupancy (queue size) is part of local state information. Thus, this second equation can be used to develop a new scheme that can be used in the Predictive Congestion Control framework. Mitra and Seery leave the details of this new scheme to the reader. Because they do not provide any specifics, the following scheme is developed.

3.1.4 Square-root queue scheme

In this subsection, the square-root scheme is first developed in the context of the framework that Mitra and Seery use and then presented as a Predictive Congestion Control scheme.

The name "square-root queue scheme" comes from the relationship between the mean nodal queue size and the window size, as Mitra and Seery derive:

$$< N >= \beta\sqrt{\lambda} + O(1).$$

The symbol $< N >$ is the mean nodal queue size, and λ is the throughput per round-trip propagation delay. In the optimal operating region, where the window size K approximately equals the throughput λ, Mitra and Seery note that

$$< N >= \beta^*\sqrt{K}; \beta^* = 1/\sqrt{M}.$$

Because the case $M = 1$, hop-by-hop control, applies to the Predictive Congestion Control framework, the queue size-window size equation becomes $< N >= \sqrt{K}$. The corresponding window control decision rule is

$$K_{n+1} = K_n - aB_n, \tag{3.1}$$

where $B_n = \sqrt{K_n} - N_n$ (a is some number $0 \leq a \leq 1$). To change the actual window size, K_n, the same conversion as before is applied:

$$\tilde{K}_{n+1} = \begin{cases} \tilde{K}_n - 1 & \text{if } K_{n+1} \leq \tilde{K}_n - 1 \\ \tilde{K}_n & \tilde{K}_n - 1 \leq K_{n+1} \leq \tilde{K}_n + 1 \\ \tilde{K}_n + 1 & \tilde{K}_n + 1 < K_{n+1} \end{cases} .$$

In words, after packet n is sent, $n = 1, 2, \ldots$, the node determines the (mean) nodal queue size N_n, calculates the bias function $B_n = \sqrt{K_n} - N_n$, adjusts the virtual window size K_{n+1}, and finally changes the actual window size $\tilde{K}_{n+1}$. If the window size is too small, $\sqrt{K_n} < N_n$, then $B_n < 0$ and the scheme increases the window size ($K_{n+1} > K_n$). By similar reasoning, if K_n is too large, the window size decreases ($K_{n+1} < K_n$).

To translate from the Mitra and Seery notation to the Predictive Congestion Control notation, $2DR_{i,out}(t)$, $\tilde{B}_i(t)$, and bias_i, $t = 0, 1, 2, \ldots$, are substituted for, respectively, the round-trip window size K_n, the mean nodal queue size N_n, and the bias term $B_n, n = 0, 1, 2, \ldots$. The output rate of node i at time t is $R_{i,out}(t)$, the quantity that the control scheme sets at time t; $\tilde{B}_i(t)$ is a moving average of the buffer occupancy $B_i(s)$ of node i for times $s \leq t$. The bias_i term is defined below. It is assumed again that $D = 1$ and that node j is the downstream neighbor of node i. With these changes, (3.1) becomes

$$2R_{i,out}(t) = 2R_{i,out}(t-1) - a\text{bias}_i, \tag{3.2}$$

where $bias_i = (2R_{i,out}(t-1))^{1/2} - \tilde{B}_i(t-1)$. When determining $\tilde{B}(t)$ (for any node), a sum $\sum_{k=0}^{N-1} b_k B(t-k)$, for some integer N, is used instead of a harder and slower to calculate integral $\frac{1}{T}\int_{t-T}^{t} B(s)ds$ for some time period T. Because $R_{out}(s)$, $s \geq 0$, is piecewise constant, $B(s)$ is continuous and piecewise linear, and hence only $B(n), n = 0, 1, 2 \ldots$, is needed for calculating $\tilde{B}(t)$. (The simulations use $N = 2$ and $b_0 = b_1 = \frac{1}{2}$.)

In the spirit of the Predictive Congestion Control framework, the state at node j at time $t+1$ is also taken into account. Node i, at time t, uses an equation similar to (3.2) to estimate the value of $R_{j,out}(t+1)$:

$$2\hat{R}_{j,out}(t+1) = 2\hat{R}_{j,out}(t) - a\widehat{\text{bias}}_j, \tag{3.3}$$

where $\widehat{\text{bias}}_j = (2\hat{R}_{j,out}(t))^{1/2} - \tilde{B}_j(t)$. Again, the $\hat{}$ notation signifies an estimate of the marked quantity, based on information known at node i at time t. The estimated output rate at time t, $\hat{R}_{out}(t)$, can be calculated in the same manner as that for the heuristic scheme, $\hat{R}_{out}(t) = \sum a_n R_{out}(t-n), \sum a_n = 1, a_1 \geq a_2 \geq \ldots \geq 0$. Since $\tilde{B}_j(t)$ depends only on $B_j(s), s \leq t$, which is known at node i at time t, $\tilde{B}_j(t)$ is not an estimate and is not marked by the $\hat{}$ notation. Because, to first order, the way to meet the performance criteria for end-to-end throughput and delay is to match input rates to output rates at each node, the control scheme strives to have $R_{j,in}(s) \approx R_{j,out}(s)$, for all s. It is noted next that $R_{i,out}(t) = R_{j,in}(t+1), t = 0, 1, 2 \ldots$, and thus $R_{i,out}(s)$ is substituted for $\hat{R}_{j,out}(s+1), s = t$ or $t-1$, in (3.3) to yield

$$2R_{i,out}(t) = 2R_{i,out}(t-1) - a\widehat{\text{bias}}_j. \tag{3.4}$$

Since the constant a is set by the control designer, (3.2) and (3.4) can be divided through by 2 on both sides and the factor of one-half absorbed into the value of a. The resulting decision rule, which combines the influences of bias_i and $\widehat{\text{bias}}_j$, is

$$R_{i,out}(t) = R_{i,out}(t-1) - a\text{Intermedval}(\text{bias}_i, \widehat{\text{bias}}_j),$$

where $\text{Intermedval}(\text{bias}_i, \widehat{\text{bias}}_j) = c_1 \text{bias}_i + c_2 \widehat{\text{bias}}_j$, and $c_1 + c_2 = 1$. (For the simulations, $c_1 = \tilde{B}_i(t-1)/(\tilde{B}_i(t-1) + \tilde{B}_j(t))$ and $c_2 = \tilde{B}_j(t)/(\tilde{B}_i(t-1) + \tilde{B}_j(t)$ are used.)

As before for the Ko, Mishra, and Tripathi schemes, to ensure that the output rate is constant between decision times, the final step in the square-root queue scheme is to take the minimum of the above "requested" $R_{i,out}(t)$ value and the flushing rate. The actual output rate is then the minimum of the two rates.

In summary, the square-root queue scheme in the Predictive Congestion Control framework proceeds as follows:

(1) Calculate $\tilde{B}_i(t-1)$, then bias_i.

(2) Estimate $\hat{R}_{j,out}(t)$, calculate $\tilde{B}_j(t)$, then estimate $\widehat{\text{bias}}_j$.

(3) Determine $\text{Intermedval}(\text{bias}_i, \widehat{\text{bias}}_j)$.

(4) Update $R_{i,out}(t) = R_{i,out}(t-1) - a\text{Intermedval}(\text{bias}_i, \widehat{\text{bias}}_j)$.

(5) Set $R_{i,out}(t) = \min(\text{opt}, R_{i,out}(t))$, where opt $= R_{i,in}(t) + B_i(t)$, the same as for the Ko, Mishra, and Tripathi schemes.

3.1.5 Dynamics of square-root queue scheme

The square-root queue scheme attempts to keep the output rate at each node equal to the square of the average buffer occupancy. However, the range of the output rate does not cover the range of the squares of the buffer occupancy. For example, in the simulations, $B_{\max}$ is 50 packets, requiring R_{out} to range up to 2500 packets per delay time unit, but the capacity of the links is only 304 packets per delay time unit.

Beyond this inherent limitation, the scheme functions as follows. At node i, for the current node bias term, if the output rate at time $t-1$ is less than (greater than) the square of the average buffer occupancy at time $t-1$, then the bias term for node i is negative (positive), causing the next output rate at time t to be greater (smaller). Intuitively, if the buffer at node i is filling up, the output rate needs to be increased to keep the buffer level stable, and if the buffer is emptying out, then the output rate does not have to be so high and can be decreased.

The influence of the next node's bias term seems counterintuitive at first glance. Again denoting the downstream node as j, if node i expects node j's buffer to begin filling up at time $t+1$, then node i increases its own output rate, contributing even more packets to node j's buffer. This appears to make node j's situation worse, not better. However, in the same situation, under the square-root queue scheme, node j increases its own output rate. Hence node i is attempting to anticipate node j's actions through the $\widehat{\text{bias}}_j$ term, with the possible advantage (or disadvantage) of lessening the magnitude of node j's changes in output rate.

3.2 Equal-risk Principle Scheme

The second new scheme proposed for the Predictive Congestion Control framework is the equal-risk principle scheme. Of the other control schemes described above, the pessimistic scheme is most similar to the equal-risk principle in philosophy. Using the equal-risk principle scheme, a node sends on a packet if it estimates that the probability of packet loss at the next node because of that node's buffer overflowing is small, that is, less than some $\epsilon, 0 < \epsilon < 1$. Using the pessimistic scheme, however, a node sends on a packet only if it is sure that there is space available for the packet in the downstream neighbor's buffer, that is, if it estimates the probability of packet loss at the next node because of buffer overflow to be exactly zero. Because of the similar philosophy behind each scheme, the control law for the equal-risk principle scheme, as developed below, ends up strikingly similar to the pessimistic control law. This similarity is discussed in detail following the description of the equal-risk scheme.

Using this equal-risk scheme, though, different nodes may send on different percentages of packets. For example, consider a virtual circuit along nodes $1 \rightarrow 2 \rightarrow 3$, with links (1,2) and (2,3) of capacity 1. Assume deterministic traffic, with the VC rate at 0.5, the cross-traffic rate on (1,2) at 0.51, and the cross-traffic rate on (2,3) at 1.0, with all traffic on a link having the same priority. Then, if $\epsilon = 0.01$, the control scheme holds back $< 2\%$ of the packets coming into node 1, but holds back >30% of the packets coming into node 2 (assuming that the cross-traffic is similarly reduced). On both links, though, there is the same risk of a packet loss because of buffer overflow. Hence, the name "equal-risk principle" (or abbreviated form "equal-risk") is given to this scheme.

3.2.1 Motivation for equal-risk principle scheme

Why equalize this risk across all the links? Consider the following back-of-the-envelope calculation. Let $H_i(l_{i-1}, l_i, b_i, X_i)$ be a function inversely proportional to the probability of packet loss at node i because of i's buffers overflowing. Assume that the function H_i is increasing in l_i and decreasing in l_{i-1}, b_i, X_i, where l_i, l_{i-1}, b_i, and X_i are,

respectively, the number of packets on i's outgoing link, the number of packets on the incoming link to i, the number of packets in i's buffer, and the number of cross-traffic packets coming into i. This captures the notion that more traffic leaving node i lowers the amount of traffic in the buffer, thus lowering the chances of the buffer overflowing, while more traffic going into or already in the buffer raises the chances of the buffer overflowing. Suppose further that the nodes keep separate buffers for each virtual circuit. Then X_i does not directly affect the buffer contents for the virtual circuit; thus, only $H_i(l_{i-1}, l_i, b_i)$ is of concern. In addition, the probability of buffer overflow at a given node i depends on the difference between l_{i-1} and l_i, not on the magnitude of l_{i-1} or l_i. Thus, attention is restricted to $H_i(l_i - l_{i-1}, b_i)$. Finally, note that the control scheme sets values for $l_1, l_2, \ldots, l_{N-1}$, where N = number of nodes in the VC. The source output rate, l_0, the destination sink rate, l_N, and the buffer occupancies, b_i, are not directly regulated by the control scheme. Hence, attention is further restricted to $H_i(l_i - l_{i-1})$ when i or $i-1$ equals $1, 2, \ldots, N-1$.

Formulating the above discussion into a maximization problem (with $N = 5$) gives

$$P : \max H_1(l_1 - l_0) + H_2(l_2 - l_1) + H_3(l_3 - l_2) + H_4(l_4 - l_3) + H_5(l_5 - l_4).$$

To maximize with respect to $l_i, i = 1, 2, 3, 4$, take the derivative with respect to l_i of the above sum and set that derivative equal to zero. The following equations result:

$$\begin{aligned} H_1'(l_1 - l_0) - H_2'(l_2 - l_1) &= 0 \\ H_2'(l_2 - l_1) - H_3'(l_3 - l_2) &= 0 \\ H_3'(l_3 - l_2) - H_4'(l_4 - l_3) &= 0 \\ H_4'(l_4 - l_3) - H_5'(l_5 - l_4) &= 0. \end{aligned}$$

Combining all of the equations into one yields

$$H_1'(l_1 - l_0) = H_2'(l_2 - l_1) = H_3'(l_3 - l_2) = H_4'(l_4 - l_3) = H_5'(l_5 - l_4).$$

Next comes a major assumption, i.e., that the behavior of the links is similar enough to use $H_i(x) = H_j(x), i, j \in \{1, 2, 3, 4, 5\}$. Thus each link has the same first-derivative value

$(H_1' = H_2' = \cdots = H_5')$. Assuming even further that H is monotonic and continuous, it is seen that equal first-derivative values for each link mean equal values for H_i. Since the probability of packet loss from buffer overflow is inversely proportional to H, the equal-risk principle follows.

Note that the equal-risk principle is derived only when $H_i = H_j, i, j \in \{1, 2, ..., N\}$. In the more general case, with heterogeneous links, the guiding idea becomes the principle of equal first-derivatives of the risks.

In the simulations, the scheme tries to minimize the loss of probability due to buffer overflows instead of maximizing some arbitrary H function.

3.2.2 Derivation of equal-risk principle scheme

Using the same notation as the Ko, Mishra, and Tripathi schemes, the derivation of the equal-risk principle scheme is presented in this section. The specific update equation follows from setting the probability of packet loss less than the given ϵ.

It is noted first that packets sent on by the current node i during $[t, t+1)$ arrive at the next node $i+1$ during $[t+1, t+2)$. To prevent the next node's buffers from overflowing during $[t+1, t+2)$, node i has to set $R_{i,out}(t)$ so that $B_{i+1}(t+2) \leq B_{\max}$. Substitute in for $B_{i+1}(t+2)$, and then for $B_{i+1}(t+1)$, using the buffer occupancy evolution formula

$$B_j(t+1) = B_j(t) + R_{j,in}(t) - R_{j,out}(t),$$

where j is any node, and the equivalence

$$R_{j,out}(t-1) = R_{j+1,in}(t),$$

where $j+1$ is j's downstream neighbor. After rearranging to leave only $R_{i,out}(t)$ on the left-hand side, the resulting equation is

$$R_{i,out}(t) \leq B_{\max} - B_{i+1}(t) - R_{i,out}(t-1) + R_{i+1,out}(t) + R_{i+1,out}(t+1). \qquad (3.5)$$

Because the last two terms in (3.5) are not known at node i at time t, their values are lower bounded in the following manner. First, (with $X_{i+1}(u)$ denoting the higher priority

cross-traffic on link $i+1$ at time u) since $R_{i+1,out}(u) \leq C - X_{i+1}(u)$ for all times u, the optimistic assumption is made that node $i+1$ sends at the highest possible rate during $[t, t+2)$, that is,

$$R_{i+1,out}(s) = C - X_{i+1}(s) \text{ for } s = t, t+1. \tag{3.6}$$

While the cross-traffic distribution is usually unknown, a Gaussian distribution is assumed to suffice for a first-order approximation, because of the large numbers of packets that pass through each node during any given time period in a high-speed, wide-area network. It is further supposed that $X_i(t), i = \text{any node}, t = 0, 1, 2, \ldots$, are i.i.d. Gaussian with mean μ and variance σ^2. Let λ denote the mean of the virtual circuit source. Then, (3.6), applied to all nodes $i+1$ and to all times s, implies that $\lambda = C - \mu$. Noting that $P(X \geq \beta) = Q((\beta - \mu)/\sigma) = \epsilon$ for $\beta = \mu + k\sigma$ and that $k = Q^{-1}(\epsilon)$, it is thus arrived at that $R_{i+1,out}(s) = C - X_{i+1}(s) \geq C - (\mu + k\sigma) = \lambda - k\sigma$ with probability $1 - \epsilon$, for $s = t, t+1$.

Therefore, if $R_{i,out}(t)$ is set to satisfy

$$R_{i,out}(t) \leq B_{\max} - B_{i+1}(t) - R_{i,out}(t-1) + 2(\lambda - k\sigma), \tag{3.7}$$

then, with probability $1 - \epsilon$, (3.5) holds true, and thus the buffer at node $i+1$ does not overflow. Thus, the foundations for the equal-risk principle scheme, described below, are set.

3.2.3 Details and dynamics of equal-risk principle scheme

From (3.7), the control equation for the equal-risk scheme is found. To maximize end-to-end throughput, $R_{i,out}(t)$ is set as high as possible in (3.7). To keep the output rate constant between decision times, the flushing rate, via the opt term, is also taken into account. The equal-risk control law is thus

$$R_{i,out}(t) = \min(\text{opt}, B_{\max} - B_{i+1}(t) - R_{i,out}(t-1) + 2(\lambda - k\sigma)), \tag{3.8}$$

where opt is defined as before. Thus, $R_{i,out}(t)$ is determined by values either known or calculable at node i at time t. Furthermore, $B_{\max}$, λ, k, and σ are set at the beginning

of the control scheme operation, and $R_{i,out}(t-1)$ is set at the previous decision time; only $B_{i+1}(t)$ has to be calculated by node i at time t.

To begin describing the dynamics of the equal-risk scheme, all of the time-independent parameters in (3.8) are combined into one constant $Z = B_{\max}+2(\lambda-k\sigma)$. Equation (3.8) becomes

$$R_{i,out}(t) = \min(\text{opt}, Z - B_{i+1}(t) - R_{i,out}(t-1)). \tag{3.9}$$

If $B_{i+1}(t)$ is held constant, (3.9) implies that if $R_{i,out}(t-1)$ is set to a value that is large relative to Z, then $R_{i,out}(t)$ must be small, and vice versa. This inversely proportional relationship could lead to oscillations. If $R_{i,out}(t-1)$ is held constant, a similar relationship between $B_{i+1}(t)$ and $R_{i,out}(t)$ is noted. The value of $B_{i+1}(t)$ is directly proportional to $R_{i,out}(t-2)$, which is inversely proportional to $R_{i,out}(t-1)$. Hence, the influence of $B_{i+1}(t)$ on $R_{i,out}(t)$ provides a balance to the influence of $R_{i,out}(t-1)$ on $R_{i,out}(t)$.

Noting that (3.9) is essentially the same as (2.1) adds further insight to the dynamics of the equal-risk scheme. Table 3.1 illustrates the traffic flow for a given arrival stream. The parameters are the same as for the virtual circuit described by Table 2.1 in Section 2.3.2. The numbers follow from the same equations that dictated the values of Table 2.1, except Z is substituted in for $B_{\max}$ in the analog for (2.3). Given $B_{\max} = 100, \lambda = 152 = \frac{1}{2}C, k = 2$, and $\sigma = 61 = \frac{1}{5}C$, $Z = 160$. Thus, for example,

$$R_{1,out}(2) = R_{2,in}(3) = \min(156 + 100, 160 - 0 - 12) = 148.$$

The discussion in Section 2.3.2 suggests that for $Z \geq 2\lambda$, where 2λ is the round-trip propagation window, there is no oscillation with the equal-risk principle. The solution to avoiding oscillations thus seems to be to set Z large enough. However, Z depends on σ, a first-order approximation of the standard deviation of the cross-traffic. For the pessimistic scheme, setting $B_{\max}$ large enough to avoid oscillations does not take the cross-traffic into account. For the equal-risk scheme, though, Z attempts to account for the variance of the cross-traffic. Because the cross-traffic may vary significantly over time

Table 3.1 Equal-risk principle scheme with $B_{max} = 100 = \frac{1}{3}C, \lambda = 152 = \frac{1}{2}C, k = 2$, and $\sigma = 61 = \frac{1}{5}C$

t	$R_{1,in}(t)$	$(\text{Loss}_1(t))$	$B_1(t)$	$R_{2,in}(t)$	$B_2(t)$	$R_{3,in}(t)$	$B_3(t)$
0	148						
1	153	(0)	0	148	0		
2	156	(41)	100	12	0	148	0
3	151	(8)	100	148	0	12	0
4	149	(139)	100	12	0	148	0
5	157	(1)	100	148	0	12	0
6	152	(145)	100	12	0	148	0
7	147	(4)	100	148	0	12	0

in practical situations, the value of Z may need to change periodically to account for any changes in the cross-traffic variance.

CHAPTER 4

SIMULATIONS

4.1 General Overview of Simulations

In the simulations that Ko, Mishra, and Tripathi perform of the static, optimistic, pessimistic, and heuristic schemes, the networks are simple, to capture the essence of the workings of the schemes.[1] They have two basic networks, pictured in Figures 4.1 and 4.2.

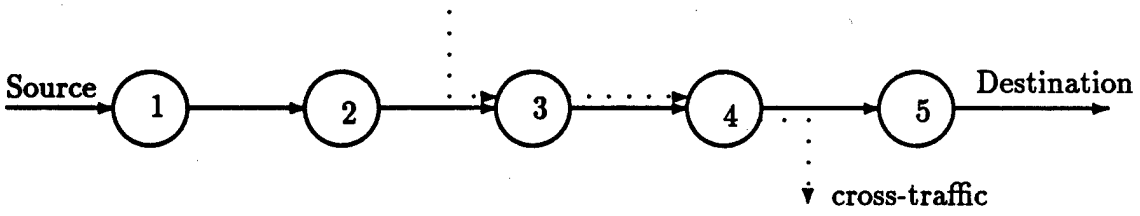

Figure 4.1 Network for 1 VC case (with cross-traffic)

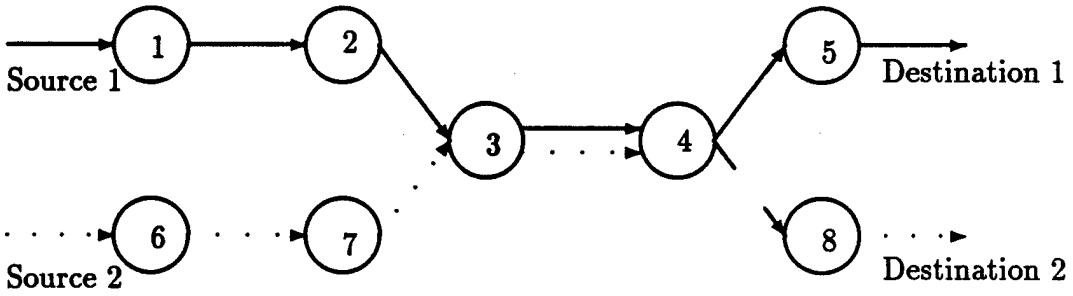

Figure 4.2 Network for 2 VC case("cross-traffic" is other VC)

Each link has capacity 1 Gb/s. Each node has $B_{\max}$ buffer slots per virtual circuit passing through it. Each link has the same one-way propagation delay $D = 10$ ms. Each packet is 4 K, or 32768 bits, long. The control decision times are one delay period

[1] See [1, 2] for full details of the Ko, Mishra, and Tripathi simulations.

apart, at $t = 0.000$ s, 0.010 s, 0.020 s, ... At each decision time, the output rate of each node is set and held constant until the next decision time. To keep the output rate constant between decision times, the value requested by each control scheme is compared to the flushing rate (announced input rate plus buffer emptying rate), and the minimum of the two rates is selected as the actual output rate. Also, the requested output rate is compared to zero and to the link capacity. These precautions prevent a node from sending, respectively, more packets than actually exist, negative packets, and more packets than can actually fit through the link. Finally, each source of packets is Poisson with mean 500 Mb/s.

The homogeneity of the networks suggests a synchronous system, even though predictive congestion control is designed to allow for both asynchronous and synchronous systems. For these simulations, the synchronicity of the network operation is exploited. Instead of having an event-driven simulation such as Ko, Mishra, and Tripathi, this simulation uses snapshots of the system at each control decision time (as used in Table 2.1 to illustrate the pessimistic scheme). The quantities of interest are the number of packets, from each virtual circuit, on each buffer and each link, as well as the number of packets dropped by each node since the most recent decision time. The buffer and link occupancies are used by the control scheme, while the dropped packets, number of arrivals (on links into nodes 1 and 6), and number of departures (on links out of nodes 5 and 8) are used to account for packets after each time step.

To allow for different values of the one-way propagation delay, the time unit is normalized to D; that is, instead of setting $R_{i,out}(t)$ at $t = 0, D, 2D, \ldots$, the control scheme sets $R_{i,out}(t)$ at $t = 0, 1, 2, \ldots$. Furthermore, since the buffer space and buffer occupancy are measured in number of packets, the rates are converted from bits/second to packets per time unit. For example, the link capacity of 1 Gb/s translates to approximately 304 packets per time unit, while the source means are all approximately 152 packets per time unit. (Note: in the simulations, a fractional number of packets is allowed.)

For a fair comparison between the Ko, Mishra, and Tripathi schemes and the two new schemes, all of the control schemes are run on the same simulation setup. For example,

the results of the pessimistic scheme from the event-driven simulation are not compared against the results of the snapshot simulation for the square-root queue scheme; instead, the performance of the pessimistic scheme in the snapshot simulation is compared to the performance of the square-root scheme in the snapshot simulation.

4.2 Simulation Results

Tables 4.1, 4.2, 4.3, and 4.4 summarize the results of the simulations, for one virtual circuit with varying intensities of cross-traffic and for two virtual circuits with different utilizations. With the one virtual circuit, the several cases in Tables 4.1 and 4.2 include the performance of the new control schemes with no cross-traffic and performance of all of the schemes against exponential, Erlang-4, and Gaussian cross-traffic with mean rate equal to one-half, two-thirds, or full capacity. The type of distribution refers to the distribution of the number of cross-traffic packets on link (3,4) during the current decision interval. The mean rate of the cross-traffic is given for each case of the simulations; in addition, the standard deviation of the Gaussian cross-traffic is one-fifth capacity. Since link (3,4) is the only link with cross-traffic, the congestion level of the network is determined by how congested this one link is. When the cross-traffic mean rate is one-half capacity, the link (3,4) capacity is almost fully utilized, hence the link is congested. When the cross-traffic mean rate is much higher than one-half capacity, e.g., at two-thirds or full capacity, the link (3,4) demand is significantly greater than the link capacity; hence, the link is not only congested, it is overloaded. (The results for the Ko, Mishra, and Tripathi schemes with no cross-traffic are included for completeness.)

Because the heuristic, square-root queue, and equal-risk schemes have adjustable parameters that affect how well the scheme performs, only the best results for the variation of the given scheme are included. For the heuristic scheme, Tables 4.2 and 4.4 list the results for proportional heuristic scheme H_{prop}.[2] In the square-root queue scheme, the value of a determines how much influence the bias terms have; Tables 4.2 and 4.4 present

[2]See [2] for details.

Table 4.1 Percent packet loss for end-to-end, network, and node 3, for control schemes for one virtual circuit case

Cross-traffic (mean & dist'n)	Control Scheme: Static			Optimistic			Pessimistic		
No cross traffic	0.61	0	0	0	0	0	0.14	0	0
$\mu = \frac{1}{2}C$									
Exponential	22.71	21.80	21.80	15.48	15.48	15.48	16.94	0	0
Erlang-4	19.56	18.68	18.68	12.39	12.39	12.39	16.00	0	0
Gaussian	15.69	14.90	14.90	10.50	10.50	10.50	13.87	0	0
$\mu = \frac{2}{3}C$									
Exponential	30.68	30.09	30.09	27.06	27.06	27.06	27.49	0	0
Erlang-4	39.61	38.86	38.86	30.91	30.91	30.91	34.13	0	0
Gaussian	37.31	36.46	36.46	34.60	34.60	34.60	35.26	0	0
$\mu = C$									
Exponential	47.05	46.36	46.36	40.01	40.01	40.01	40.36	0	0
Erlang-4	65.34	64.54	64.54	61.35	61.35	61.35	61.75	0	0
Gaussian	84.52	83.80	83.80	82.86	82.86	82.86	83.44	0	0

Table 4.2 Percent packet loss for end-to-end, network, and node 3, for control schemes for one virtual circuit case

Cross-traffic (mean & dist'n)	Control Scheme: Heuristic			Square-root Queue			Equal risk		
No cross traffic	1.13	0	0	4.42	4.21	1.24	0	0	0
$\mu = \frac{1}{2}C$									
Exponential	24.12	3.93	1.63	18.09	17.88	16.38	16.77	16.77	16.77
Erlang-4	17.37	4.40	1.79	15.58	15.43	13.71	12.63	12.63	12.63
Gaussian	10.93	2.94	1.10	13.60	13.47	12.00	8.92	8.92	8.92
$\mu = \frac{2}{3}C$									
Exponential	33.85	4.46	1.70	28.30	28.22	25.95	25.12	25.12	25.12
Erlang-4	35.94	3.95	1.24	35.17	35.08	33.00	28.20	28.15	28.15
Gaussian	36.20	3.68	1.11	35.24	35.00	33.31	33.69	33.65	33.65
$\mu = C$									
Exponential	47.19	3.70	1.26	46.09	45.98	42.65	42.35	42.35	42.35
Erlang-4	61.87	2.13	0.45	65.18	64.98	61.23	61.61	61.51	61.51
Gaussian	80.10	0.87	0.05	85.52	85.46	83.91	84.66	84.30	84.30

Table 4.3 Percent packet loss for end-to-end, network, and node 3, for control schemes for two virtual circuits case

VC # (mean)	Control Scheme: Static			Optimistic			Pessimistic		
$(\lambda = \frac{1}{2}C)$									
VC1	0.68	0	0	0.41	0.41	0.41	0.48	0	0
VC2	0.85	0	0	0.42	0.42	0.42	0	0	0
$(\lambda = C)$									
VC1	49.96	49.10	49.10	49.86	49.13	49.13	50.01	0	0
VC2	49.86	49.16	49.16	49.63	49.07	49.07	49.82	0	0

Table 4.4 Percent packet loss for end-to-end, network, and node 3, for control schemes for two virtual circuits case

VC # (mean)	Control Scheme: Heuristic			Square-root Queue			Equal risk		
$(\lambda = \frac{1}{2}C)$									
VC1	1.60	0.04	0	1.78	1.53	1.07	0.05	0.02	0.02
VC2	1.78	0.05	0	2.34	1.96	1.04	0.24	0.01	0.01
$(\lambda = C)$									
VC1	50.38	0.01	0	49.89	48.78	48.69	49.67	49.20	49.20
VC2	50.58	0.01	0	49.99	48.86	48.81	49.71	49.06	49.06

the results for $a = 0.125$. The value of k in the equal-risk scheme depends on what value of ϵ the control designer wants; the results in Tables 4.2 and 4.4 are for the case $k = 2$ (or $\epsilon = 0.02274$).

One general note for the cross-traffic generation in the simulations with one virtual circuit concerns the cases with higher cross-traffic means. If the mean of the cross-traffic is near or equal to capacity, the number of cross-traffic packets is generated according to the given distribution, but then the simulation truncates that number to be less than or equal to C. As a result, the actual mean of the cross-traffic on link (3,4) is less than the value given; the difference is greater for random variables with higher variance. For example, in Tables 4.1 and 4.2, with the cross-traffic mean equal to capacity, all schemes perform worst against Gaussian cross-traffic, which has the lowest variance, and best against exponential cross-traffic, which has the highest variance. These results imply that the actual Gaussian mean is close to capacity (thus link (3,4) is almost fully utilized by the higher-priority cross-traffic), while the actual exponential mean is significantly lower than capacity (thus a much higher number of VC packets can pass through link(3,4)).

The performance measures are three packet loss percentages: end-to-end percent loss, network percent loss, and node 3 percent loss. All three measures are calculated as a percentage of the total number of arrivals to the system. The percentages of packet loss in various sections of the network are used since a low percentage of packet loss in a given section implies that the average throughput of the virtual circuit in that section is high. The end-to-end loss is the percentage of arrivals dropped at any node in the VC; the network loss is the percentage of arrivals dropped after node 1, which serves as access control for the VC; and the node 3 loss is the percentage of arrivals lost at node 3 because of congestion on link (3,4), the only link subject to cross-traffic. For example, in Table 4.1, for the pessimistic scheme, with the exponential cross-traffic mean equal to one-half capacity, 16.94% of the arriving packets are lost from end-to-end, 0% are lost after node 1, and 0% are dropped at node 3 (all of which supports the claim that the pessimistic scheme drops packets only at the first node). Another example is the results for the square-root scheme, with the Gaussian cross-traffic mean equal to

capacity, in Table 4.2; the corresponding end-to-end loss, network loss, and node 3 loss are, respectively, 85.52%, 85.46%, and 83.91%.

These numbers all include the effect of the first node's control decisions. Since this node is the first to regulate the arriving packets, it acts as both congestion control and access control, to relieve congestion in the network and to allow or deny packets admission into the network. The difference between the end-to-end loss and the network loss reflects how well the scheme performs as access control. For example, the pessimistic scheme in Tables 4.1 and 4.3 and the heuristic scheme in Tables 4.2 and 4.4 have large differences between the end-to-end loss and the network loss, reflecting that the first node exerts a strong admission control. On the other hand, there is the optimistic scheme in Table 4.1, which has no difference between the end-to-end loss and the network loss; the optimistic scheme applies no admission control, and the results illustrate that fact. However, there is more interest in these schemes as congestion control, that is, how the schemes handle the packets once the packets are in the network. Dividing the network loss and the node 3 loss by the quantity one minus the difference between the end-to-end loss and the network loss removes the effect of the admission control. The adjusted loss measures then show what fraction of packets are lost, in the network or at the congestion point, once they have entered the network. For example, with Gaussian cross-traffic having a mean of one-half capacity, the heuristic scheme's 2.94% network loss and 1.10% node 3 loss in Table 4.2 become 3.20% and 1.20%, respectively, after dividing by 1-(0.1093 - 0.0294). The adjusted network loss of 3.20% means that 3.20% of the packets sent on by the first node are dropped later along the virtual circuit; that value does not mean that 3.20% of the arrivals are dropped after node 1 in the virtual circuit. Note that, since the end-to-end loss is always greater than or equal to the network loss, the adjusted loss rates are always greater than the original loss rates.

For the two virtual circuits simulations, each VC has the same control scheme. The various cases in Tables 4.3 and 4.4 include congested levels of traffic (λ = mean of the VC source = one-half capacity for both VC's) and severely overloaded levels (λ = capacity for both VC's). If the total requested output rate $R_{i,out,1}(t) + R_{i,out,2}(t)$ is greater than

C, then the output rates are scaled back by a common factor to fit into the link capacity. This scaling factor is $C/(R_1 + R_2)$, so that $[C/(R_1 + R_2)] * (R_1 + R_2) = C$. ($R_n$ is shorthand for $R_{i,out,n}(t)$, $n = 1, 2$.)

Note that given the same possible link (3,4) utilization, the control schemes perform much better for the two virtual circuits case than for the equivalent one virtual circuit case. For example, compared to the entries in Tables 4.1 and 4.2, for the case $\mu = C/2$ for all cross-traffic distributions, the corresponding entries in Tables 4.3 and 4.4, for the case $\lambda = C/2$, are much lower in magnitude. Even when the link (3,4) utilization is much higher for the two virtual circuits case, the schemes can perform much better. Given two virtual circuits with $\lambda = C$, note that link (3,4) has $\rho \approx 2$, which is higher than the $\rho \approx 1.5$ for one virtual circuit with cross-traffic $\mu = C$. The tables show that all of the schemes have higher losses against Gaussian and Erlang-4 cross-traffic than against the other controlled virtual circuit.

The two main causes for the differences between the one virtual circuit and two virtual circuits case are the priority assignments and the variance of the "cross-traffic." For one virtual circuit, the cross-traffic has higher priority than the virtual circuit traffic, while for two virtual circuits, the two virtual circuits have equal priority. Hence a given cross-traffic rate takes a bigger chunk of bandwidth from a virtual circuit than does a competing controlled virtual circuit with the same rate. In addition, the controlled virtual circuits have less fluctuation in the level of traffic within the network than do the random cross-traffic distributions; hence, one virtual circuit can predict the future behavior of another controlled virtual circuit better than it can estimate the future behavior of random cross-traffic.

For the above reasons, in the two virtual circuit simulations, the results in Tables 4.3 and 4.4 are all nearly the same. For congested levels, all schemes lose few packets, usually <1% (with only the heuristic and square-root queue schemes having losses in the 1-2% range, as shown by Table 4.4). For severely overloaded levels, the losses are all approximately 49-50%, except for the pessimistic and heuristic schemes. The pessimistic scheme has no packets lost inside the network (hence network loss and node 3 loss are

both 0 in Table 4.3). The heuristic scheme also has a low network loss of 0.01% and a low node 3 loss of 0%, as listed in Table 4.4.

CHAPTER 5

CONCLUDING REMARKS

5.1 Conclusions

Compared to the benchmark of the static scheme performance results, all of the other schemes described above perform better except in some notable cases. The heuristic scheme has a higher end-to-end loss for one virtual circuit with no cross-traffic and for two virtual circuits with λ equal to one-half capacity. This exception implies only that the heuristic scheme is much more stringent with arrivals than is the static scheme, which is vacuously true because the static scheme imposes no control over incoming packets. Also, in the cases of one virtual circuit with no cross-traffic and of two virtual circuits with λ equal to one-half capacity, the square-root queue scheme has higher losses across the board. This exception is more serious, because it suggests some fundamental flaw in the square-root queue scheme as it is simulated.

The best schemes, judged by performance criteria, are the pessimistic and heuristic schemes. If the simplicity of the scheme influences the ranking, then the pessimistic scheme is best, followed by the heuristic scheme, because the pessimistic scheme is less complex than the heuristic. Similarly, though the two new schemes perform comparably to the static and optimistic schemes, the utter simplicity of the static and optimistic schemes tilt the scale in their favor. In fact, if the high speed of the network demands an extremely simple control scheme, the optimistic scheme is the best scheme, since it performs as well as or better end-to-end than do the pessimistic and heuristic schemes. The main drawback is that no backpressure exists; if a link in the virtual circuit fails, the scheme has no way in which to compensate and still sends incoming packets to the failed link.

The two new schemes perform as well as the static scheme, except as noted above. However, the added complexity of these two schemes makes them less desirable than the other schemes developed by Ko, Mishra, and Tripathi.

5.2 Further Areas of Research

Three main questions remain to be answered. One which this thesis attempted and failed to answer is "Are there better control schemes for the Predictive Congestion Control framework?" Perhaps the motivation behind the square-root queue and equal-risk schemes contains the seed for better control schemes than the four Ko, Mishra, and Tripathi schemes. For example, simply varying the parameters in the new control schemes may affect how they perform. In the square-root queue scheme, different weightings of bias_i and $\widehat{bias}_j$, better averaging methods for $\tilde{B}(t)$, and better estimating techniques for $\hat{R}_{j,out}(t)$ may improve the performance of the scheme. For the equal-risk scheme, the other distributions may serve as better first-order approximations of the cross-traffic, especially if the network provides some information about the second and higher moments of the cross-traffic.

The second question, related to the first, concerns how much better a control scheme can be, within the Predictive Congestion Control framework: "Given the restrictions of the Predictive Congestion Control framework, what is the best performance over all control schemes?"

The third question addresses an issue outside the scope of this thesis. Ko, Mishra, and Tripathi developed the Predictive Congestion Control framework to handle the needs of high-speed, wide-area networks. Just how fundamentally sound is the Predictive Congestion Control approach?

REFERENCES

[1] K.-T. Ko, P. P. Mishra, and S. K. Tripathi, "Predictive congestion control in high-speed wide-area networks," submitted for publication, October 1990.

[2] K.-T. Ko, P. P. Mishra, and S. K. Tripathi, "Interaction among virtual circuits using predictive congestion control," presented at 2nd VHSN Workshop, Greenbelt, MD, March 1991.

[3] D. Mitra and J. B. Seery, "Dynamic adaptive windows for high speed data networks: theory and simulations," technical report, AT&T Bell Laboratories, May 1990.